Good work, Patrick! I'm glad to know that this algorithm is implementable in practice by an E27 student.

Grade: A

# FINAL PROJECT: EULERIAN VIDEO MAGNIFICATION

## MOTIVATION

Keeping track of and noticing small and subtle changes in an environment can be extremely useful for a large variety of purposes. For example, if we consider two different bridge structures under the same type of stress, to the naked eye, it is almost impossible to tell which bridge is more lenient to the stress, unless one of the bridges literally breaks down. Another example is where we would need to monitor the vital signs of a new born baby with non invasive methods so as to not disturb the infant's bodily functions and to prevent harm.

Eulerian video magnification presents methods to solve these problems. For example, magnifying the motion of the recordings of the two bridges under the same type of stress can allow us to clearly see which bridge structure is more lenient or sturdy. Similarly, magnifying a video of a baby sleeping can enhance the movement in the video of the baby's torso heaving up and down. We can then take the peak of this motion as the starting point to measure the infant's breathing pattern to spot any irregularities in their vital signs.

## SOLUTION

Consider a video and all of its respective frames. We take the spatial decomposition of each of these frames by forming a laplacian pyramid of each of the frames. We then apply a temporal filter to all of the frames with a specified low and high frequency range of interest. These frequencies are then amplified, and added back to the original frames. The frames are now amplified only with respect to the specified frequency range of interest. When we then recollapse the laplacian pyramid, and play all the frames as a video, what we get is a video where motion that falls into the specified frequency range is amplified. Since our interest is in trying to amplify subtle changes, but not noise, the desired frequency range usually lives in lower frequencies.

The only part that I was completely unfamiliar with was the part where I had to select out a range of desired frequencies to amplify them and add them to the laplacian pyramid. I took inspiration from **https://www.programcreek.com/python/example/59508/scipy.signal.butter** to implement the butterworth bandpass filter using the scipy module.

## CODE

I wrote only one python file named eulerian_video_mag.py in the repository which contains a series of functions dealing with loading and saving video files, building and reconstructing a laplacian pyramid for a video, and signal processing. Originally, I wanted to implement both the eulerian video magnification for motion and color, but I only ended up being able to do motion amplification with the time constraint. I was a little confused about the way in which I was supposed to implement color magnification, as I needed to implement a temporal bandpass filter instead of a butterworth bandpass filter.

There were a total of 8 functions in the file.

1. magnify_motion() - This is the main function outlining the entire process. It takes in a video as the input, along with the low and high frequency cut offs. Then, it builds the laplacian pyramid for all the frames in the video, then applies a butterworth bandpass filter to extract the frequency range of interest. We then amplify these frequencies, which is set to be 20, but can be set higher if the changes are still too subtle to be properly noticed. We then reconstruct the original video by recollapsing all the laplacian pyramids of all the frames. Then we save the video as the output.
2. load_video() - This takes in the video and uses open cv functions to put all the frames into a list. It returns the list of frames and the frames per second.
3. save_video() - This takes in the reconstructed video and just saves it as the output in the current working directory as “out.avi”.
4. pyr_build() - This function was reworked from Project 2 because the prior implementation was not my best. It takes in a frame, and creates a gaussian pyramid of the image. Since we know that the laplacian pyramid is just the difference between the successive gaussian blurs, we can index consecutive elements of the gaussian pyramid list and find the difference between them. It then returns the laplacian pyramid as a list.
5. laplacian_video() - I finally got to implement laplacian pyramid for a video as I did not have time to do it as an extension for the second project. It takes in a video and creates a laplacian pyramid for all the frames in the video by calling pyr_build() multiple times. It then returns the doubly indexed list containing the laplacian pyramid for all the frame sin the video. **There is a consideration that I am going to skip for now that concerns both this function and pyr_build() that is discussed in the limitations and extensions concerning the depth of the spatial decomposition.*
6. butter_bandpass_filter() - It takes in the frame, the frames per second, and the low and high caps for the desired frequency range. This function considers the first few levels of the spatial decomposition and returns the non amplified frequency range of interest.
7. pyr_reconstruct() - This function takes in a list of the filtered frequencies for all the frames, and adds the amplified frequencies to the original pyramid layers. It then collapses the layers to reconstruct all the respective frames in the video, and returns the list of reconstructed frames.
8. main() - This takes care of command line arguments. The expected input is shown if the wrong input is given. The expected input is as follows:
    a. python eulerian_video_mag.py video_name low high

    where low and high represents the low frequency and high frequency cut offs for the desired range.

**OUTPUTS**

I chose low = 0.5 and high = 3 as this seemed to work the best. If I chose the low frequency too low, then it considers random noise as the desired frequency and noise is largely amplified. There is no need for the high cutoff frequency to be high as this defeats the purpose of eulerian video magnification. If the frequency is already really high then the change is not subtle and there is no need to magnify the motion.

There was a lot of noise in the beginning of the video, which was present in the undesired white outputs in the first few seconds of the magnified video. However, this went away afterwards.

I also tried changing the amplification factor from 10 to 30 and the results are as shown. It seems like 10 is a better value as 30 amplifies too much of the noise as well. I cannot put the actual outputs here as they are videos but these snapshots below show the differences between a normal video and after motion magnification:

Amplification = 10

Amplification = 30

The video of this baby was taken from the website http://people.csail.mit.edu/mrub/evm/. I also tested another video of testing the strength of a small bridge. The video was taken from https://www.youtube.com/watch?v=rTdQL2KeTn0.

**LIMITATIONS AND EXTENSIONS**

I am unsure why but if I create a laplacian pyramid as in project 2 where we built the pyramid until only a couple of pixels remained, up to the appropriate depth, the frequency magnification no longer works as well. I decided to reduce the depth to 3 as this seemed to work the best.

One of the limitations as discussed in the previous section was noise. In the beginning of the video, the noise was immensely amplified, causing undesirable white outputs in the first few seconds of the video, but went away afterwards. Perhaps there is a method parallel to gaussian blurring to reduce noise in images for videos.

Another limitation that came to mind was concerning color magnification which I did not implement, but did some research on. One of the applications of this method was that it was a non-invasive method of monitoring blood flow in different organisms or even humans. However, different skin tones can present difficulties in the ability to see blood flow clearly from a video. One thought that came to mind was relative standardization. If we consider the Normal Distribution, we can always standardize it by subtracting all the data points with its mean and dividing it by the standard deviation to get the Standard Normal Distribution. Perhaps variations between skin color and blood flow is relative, but more subtle in other skin tones so we could compare the relative changes and conclude information about blood flow based off of available data. This is just a thought and would require a lot more research to see if it is a thing people do or not.

If I had more time, I would love to dive into the relationship between live eulerian video magnification and homographies. Obviously, without perspective consideration, moving your camera in live video magnification is going to amplify almost all the pixels as they are all changing. However, if we consider the relative motion from frame to frame, perhaps it would be possible to map the current frame to the previous using a homography and only amplify the changes that occur there. I would also try out the code on a bunch of more video inputs as it is always fun to see what subtle changes exist in random videos.

Most of the research was done here:
https://www.youtube.com/watch?v=ONZcjs1Pjmk
https://core.ac.uk/download/pdf/78051554.pdf
http://people.csail.mit.edu/mrub/evm/